Barış Ayyıldız 1901042252

1) $\sum_{i=1}^{n} c = cn \in \Theta(n)$

Homework-3 Divide & Conquer Approach:

$T(n) = 2T\left(\frac{n}{2}\right) + \underbrace{M(n)}$ → cross sum

$M(n) = \sum_{i=1}^{n} 1 + \sum_{i=1}^{n} 1 = 2n$

$\Rightarrow T(n) = 2T\left(\frac{n}{2}\right) + 2n$ from Master Theorem $a=2$ $b=2$

case-2 → $2n = \Theta(n^{\log_2 2} \log^k n)$

$T(n) \in \Theta(n^{\log_2 2} \log^{k+1} n)$

$T(n) \in \Theta(n \log n)$

Dynamic programming solves the problem more efficiently.

CamScanner ile tarandı

2) knapsack method : ( $n \to$ size of input
$m \to$ cap size )

$$T(n) = \underbrace{\sum_{i=1}^{m+1} \sum_{j=1}^{n+1} c}_{\text{table creation}} + \sum_{i=1}^{n+1} \sum_{j=1}^{m+1} c = \sum_{i=1}^{m+1} (n+1)c + \sum_{i=1}^{n+1} (m+1)c$$

$$T(n) = (n+1)(m+1)c + (n+1)(m+1)c$$
$$T(n) = (mnc + nc + mc + c).2 \Rightarrow T(n) \in \Theta(nm)$$

3) knapsack method : ( $n \to$ size of input
$m \to$ cap size )

$$T(n) = \text{insertionSort}(n) + \sum_{i=1}^{n} c = \text{insertionSort}(n) + n$$

worst case : Insertion sort takes $\Theta(n^2)$ time

$$T(n) = n^2 + n \in \Theta(n^2)$$

Best case : Insertion sort takes $O(n)$ time

$$T(n) = n + n \in O(n)$$

4) maxCourses method:

$$T(n) = \text{insertionSort}(n) + \sum_{i=1}^{n} c = \text{insertionSort}(n) + n$$

worst-case $\rightarrow T(n) = n^2 + n \in \Theta(n^2)$,

best-case $\rightarrow T(n) = n + n = 2n \in \Theta(n)$,